Docket No. P18182 UTILITY PATENT

UNITED STATES APPLICATION FOR LETTERS PATENT

for

DISTRIBUTED EXTERIOR GATEWAY PROTOCOL

by

Sanjay Bakshi and Rajendra S. Yavatkar

filed

November 13, 2003

DISTRIBUTED EXTERIOR GATEWAY PROTOCOL

BACKGROUND

The Internet is an example of a collection of autonomous systems that define the administrative authority and routing policies of different organizations. Autonomous systems run interior gateway protocols within their boundaries and interconnect via exterior gateway protocols. A gateway here is defined as any entity that allows other devices to gain entrance to the network. The boundaries could be a domain or other sub-network defined by a policy or an enterprise. Examples of an interior gateway protocol may include Open Shortest Path First (OSPF), Interior Gateway Protocol (IGP), and Enhanced Interior Gateway Protocol (EIGP), as examples. Exterior gateway protocols may include Border Gateway Protocols (BGP).

BGP was introduced because the interior protocols typically do not scale beyond a particular enterprise. Border gateways exchange network reachability information with other border gateways, and use BGP to do so. This reachability information about the autonomous systems within a domain allows the border gateways to construct a graph of autonomous system connectivity from which routing loops can be pruned and allows some policy decisions at the system level to be enforced. BGP manages these communications and is run over a reliable transport protocol, such as the Transmission Control Protocol (TCP), between the two gateways exchanging information.

The initial exchange of information generally involves the two devices exchanging entire routing tables, with periodic updates sent to modify the routing table to add or delete entries. A major problem with the current implementation of BGP is scalability in the face of heavy routing churns. During the Code Red 1, Code Red 2 and Nimda worm attacks, these update messages increased to over 6.67 million from an average of 2 million normally seen. The currently deployed BGP software, where all the BGP functionality is processed by a

central processor in each device caused the processors to become overwhelmed and caused BGP session resets. In addition, the TCP KEEPALIVE messages that indicate which devices are still operating were not delivered, also leading to BGP sessions to be reset.

In addition to these issues, BGP gateways suffer from security vulnerabilities, such as bogus TCP SYN messages, referred to as TCP SYN attacks, tampered BGP UPDATE messages announcing an illegal or invalid next hop, etc. These types of attacks succeed because the centralized architecture of BGP devices require these messages to be processed by the central processor which quickly becomes overwhelmed, denying processing of legitimate requests from other devices.

BRIEF DESCRIPTION OF THE DRAWINGS

Embodiments of the invention may be best understood by reading the disclosure with reference to the drawings, wherein:

Figure 1 shows an example of border gateways between two domains.

Figure 2 shows an embodiment of a border gateway device having a distributed architecture.

Figure 3 shows a flowchart of an embodiment of a method to process BGP traffic.

Figure 4 shows a flowchart of an embodiment of a method to initialize a control card in a BGP device.

Figure 5 shows a flowchart of an embodiment of a method to initialize an offload card in a BGP device.

DETAILED DESCRIPTION OF THE EMBODIMENTS

Figure 1 shows an example of border gateways between 2 domains. These gateways provide connection for the domain to other domains, communicating with other border gateways, also referred to as gateway peers. These gateway peers communicate via an exterior gateway protocol, such as the Border Gateway Protocol. Examples used here may be based upon BGP, but are applicable to any exterior gateway protocol running on a reliable

transport protocol. A reliable transport protocol, such as Transmission Control Protocol (TCP), has mechanisms that verify and validate data transmissions, eliminating the need for update fragmentation, retransmission acknowledgement and sequencing. Any authentication scheme used by the transport protocol may be used in addition to the authentication mechanisms used by the exterior gateway protocol.

The gateways 10 and 12 exchange messages to open and confirm connection parameters. Incremental updates are sent as the routing tables change. Exterior gateway protocols may not require periodic refreshes of the entire routing table. A device communicating in this protocol must maintain current versions of all of its peers for the duration of the connection. KeepAlive messages and updates are sent periodically to ensure the liveness of the connection and the accuracy of the routing tables.

The updates are typically used to exchange routing information between two border gateways and to announce new routes or modify/delete previously announced routes. These updates are currently processed by the central processor on a gateway. During attacks, such as the Nimda, Code 1 and Code 2 worm attacks, the border gateways were overwhelmed with the update message traffic and could not process them. This causes 'session resets' or connections to be closed, which typically happens when a border gateway encounters an error condition. In addition, the high level of processing required for all of the updates caused the processor to fail to send out KeepAlive messages. The lack of the KeepAlive messages caused connection timers to time out. As a result, other devices closed the connections to the flooded devices that were not transmitting the KeepAlives. The closure of all of these connections then lead to 'denial of service' as legitimate users attempted to access devices that were not longer 'talking to each other.'

Other vulnerabilities exist in the current implementations of border gateways. In TCP, for example, a session is opened by a TCP SYN message (synchronization or start) being received at a destination host. Typically the destination host sends a SYN ACK or

acknowledgement message to the source host. In addition, the destination host opens a queue of finite size for a predetermined period of time to keep track of connections waiting to be completed. Upon receipt of an ACK message acknowledging SYN ACK, the connection is established. The queue of connections awaiting completion generally empties very quickly as the ACK messages are usually received in a matter of milliseconds, while the timer on the queue may be set to expire in a minute. This sequence is sometimes referred to as the 'TCP three-way handshake.'

In a TCP SYN attack, a destination host is flooded with TCP SYN messages that have invalid addresses. An invalid address is one that does not have any device associated with it. The destination host sends out SYN ACK messages to the bogus or illegal addresses and opens the connection queue. Since the address of the source is invalid, the connection queue remains open for the full period of the timer. If enough invalid TCP SYN messages are sent, the processor becomes overwhelmed managing the connection queues and service is again denied legitimate users.

Other types of attacks include mal-formed packets and tampered update messages that announce either an invalid or an illegal next hop for a valid destination or set of destinations. A mal-formed packet in this context is one that is mal-formed with respect to the protocol. Some mal-formed packets are caught by computing the checksums, which does not catch mal-formed packets with respect to the protocol. An illegal next hop is one where the address is of someone trying to gain access to the data in a packet illegally, 'pirating' the traffic. All of these attacks can be avoided or handled by packet filters to identify mal-formed packets, address filters to identify bogus addresses, etc. However, these filters require processing cycles and resources that become scarce during attacks.

Embodiments of this invention propose a network architecture that enables many of the packet processing tasks to be offloaded onto line cards. In addition, maintenance of the state machine for connections, as well as policy decisions, can be moved off the central

processor and distributed among several line processors. An example of such a device is shown in Figure 2.

The network device or element of Figure 2 may be a physical device, or a collection of devices connected by a backplane of network connections. In the context of the above discussion, this device would be a gateway running an exterior gateway protocol such as BGP. The device 10 has a control card 20 that has a central processor 22 and a storage 24 for storing the routing table of the device and those of its peers. The central processor is more than likely a general-purpose processor, such as an Intel® Architecture (IA) processor of the Pentium® or higher class. The control card 20 communicates through the backplane 36 through the appropriate port or interface 26. The control card is connected through the backplane 36 to a number of line cards, such as 30. There will more than likely be several line cards, the use of just one is just for ease of discussion. The line card 30 has a processor 32 and a port 34 that allows it to terminate the communications coming into the device through the reliable transport protocol. The line processor is a network-enabled processor, optimized for handling network traffic, such as an Intel® IXP Processor (Intel Exchange Processor). A network-enabled processor has a general-purpose processor and at least one microengine, such as a reduced instruction set computer (RSIC) processor that may be used for packet processing.

The incoming data is processed and handled by several line cards that all have network-enabled processors. This allows the processing to be distributed and avoids overloading the central processor or any of the line processors. The incoming data is received at the line card and only valid update data is sent to the control card. The control card then performs all of the central protocol processes, such as maintaining the routing table and the configuration information of the individual line cards.

The line cards also handle generation of outgoing messages, either in response to incoming messages or other events, such as a routing table change, encryption and decryption

of packets, caching local copies of the routing table, filtering mal-formed, illegal and invalid packets, as well as parsing and validating incoming packets. In addition, the output policy engine that determines to which peers what messages are sent, and the BGP state machine that maintains the status of all of the connections can be offloaded to the line cards. All of these functions are referred to as performing exterior gateway protocol or border gateway protocol BGP functions. The portions of the protocol that have been offloaded are referred to as the offload portion, and the portion retained by the control processor will be referred to as the control portion. This distribution of the processing tasks allows the line cards to screen the incoming traffic before it is sent to the central processor. An example of this process is shown in the flowchart of Figure 3.

At 40 the traffic across the reliable transport connection is received at the line card, as the line card is now the termination of this connection. At 42 messages are examined to determine if they are valid. If they are not valid, they are not processed, either being discarded or otherwise disposed of and not sent to the control processor at 50. If the packets are valid, the line cards parse the packets and extract the information, such as an update at 44. The valid data is then transmitted to the control card at 46. Any outgoing messages to peer gateways, such as changes in the routing tables, changes in policy status, etc. are transmitted at 48.

Tracking of which line cards are performing what processes on which packets can be handled in several ways. An architecture that enables this distribution of processing may be referred to as Distributed Control Plane Architecture (DCPA). This is discussed in more detail in co-pending patent application US Patent Application No. 10/XXX,XXX, "Distributed Control Plane Architecture for Network Elements," filed November 14, 2003. The DCPA is exemplified by a DCPA Infrastructure Module that contains the logic required by a control card or line card to discover control cards or line cards present in a system, establish connectivity with them, and exchange information about their capabilities. The

DIM would execute on both the control card and the line card in Figure 2. The use of such architecture allows the distribution of processing to occur.

In addition, a virtual interface may be used to receive the incoming packets and determine to which entity the packet should be routed. An example of such an interface is shown in co-pending patent application, "Implementation Of Control Plane Protocols And Networking Stacks In A Distributed Network Device," US Patent Application No. 10/XXX,XXX, filed November 14, 2003. The use of architectures and interfaces such as these allow the distribution process to remain coordinated.

This can be seen in the embodiments of methods to initialize the control card and line cards shown in Figures 4 and 5. In Figure 4, a line card prepares to handle distributed BGP processing. At 60, the line card initializes, such as 'booting up.' The line card then registers itself at 62 with a central registration point, such as the DIM of the DCPA. This allows the device to track the capabilities of the line card and its processing resources available, as well as providing the line card with a means to identify which events it may have an interest.

At 64, the line card determines if the control card has registered. When the control card has registered, being an event for which the line card will more than likely have registered, the two entities set up a control connection at 66. The line card transmits its resource data as to which interfaces it has on it at 68, as well as its processing resources at 72. The control card then provides the configuration information to the line card at 70. At 72, the line cards establish the peer connections with other border gateways and begin accepting and processing packets. The line cards then perform the BGP functions mentioned above at 74 and transmit valid data to the control card at 76 as necessary. These last two processes repeat until the device is shut down.

On the other side of the process, the control card is prepared for distributed processing according to a process such as the one shown in Figure 5. The control card is initialized at 80 and registers with a central register such as the DIM at 82. If the line cards are not registered

yet, the control card waits until they are. Once the control cards and the line cards have registered, they discover each other and establish a control connection at 86. The control card then provides configuration information at 88, such as the BGP routing table, and policy data to allow the line card to make output policy decisions about to which peers messages need to be transmitted. This effectively offloads the BGP functions mentioned above to the line cards, leaving the central processor to perform the central BGP functions at 90.

It must be noted that network devices already exist that have a control card and line cards with processors on them. Converting these machines to distributed BGP gateways would be a matter of upgrading the software instructions that direct the operation of the machine. In this instance, the embodiments of the invention may be implemented as an article of machine-readable instructions that, when executed, cause a machine to perform the methods and processes of the invention.

For example, an existing border gateway that is running BGP over TCP and has line processors with the necessary capacity could become a border gateway implementing distributed BGP. All that would be needed would be an upgrade to the operating software of that gateway.

Thus, although there has been described to this point a particular embodiment for a method and apparatus for distributed exterior gateway protocols running over reliable transport protocols, it is not intended that such specific references be considered as limitations upon the scope of this invention except in-so-far as set forth in the following claims.